Attorney Docket No. : ROC920020009US1
PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: **Lucck, David** Serial No. **10/697,503** Filed: **10/30/2003** For: **"Pipeline Recirculation for Data Misprediction in a Fast-Load Data Cache"**	Customer Number: **55992**

AFFIDAVIT PURSUANT TO 37 C.F.R. 1.131

1. My name is David A. Luick. I am over twenty-one years old and I make the following declaration based on my own personal knowledge.

2. I am the inventor of the invention disclosed and claimed in the above-referenced patent application, which I have assigned to IBM Corporation (hereinafter "IBM").

3. I conceived and reduced to practice the above-referenced invention as part of my work for IBM in Rochester, Minnesota, by no later than March 31, 2003. This is evidenced by the document entitled "Disclosure ROC8-2001-0454" attached hereto as Exhibit A.

4. The document shown in Exhibit A is a standard IBM invention disclosure form. It contains a complete disclosure of the invention claimed in the present application.

5. Upon reducing the invention to practice, I completed this invention disclosure document and submitted it to the IBM Office of Counsel, Intellectual Property Law by August 15, 2001. This is also evidenced in Exhibit A.

6. Typically, invention disclosure submissions at IBM go through a review process in which an invention disclosure team (IDT) reviews a disclosure to determine if it meets IBM's criteria for pursuing patent protection. Because of the number of tasks involved in the review and the case load at IBM, this review typically takes several months.

7. If the IDT recommends pursuing patent protection for the disclosed invention, then the IBM Office of Counsel, Intellectual Property Law makes a final decision on whether to pursue the application and, if the decision is to pursue patent protection, it transmits a

copy of the disclosure to a patent attorney or agent to prepare the patent application.

8. The final decision in the present case was made by January 15, 2002 and the disclosure, which was then assigned Docket No. ROC920020009US1, was sent to Bryan W. Bockhop, Esq. on that date. This is evidenced in a letter sent to Mr. Bockhop on that date, attached hereto as Exhibit B.

9. U.S. Patent Application No. 10/697,503, [redacted], was subsequently filed on October 30, 2003.

10. Each responsible person was diligent in pursuing patent protection for this invention at each stage of the disclosure approval process and the patent application preparation and filing process.

I declare under penalty of perjury that the foregoing is true and correct.

July 25, 2007
Date

David A. Luick

Disclosure ROC8-2001-0454

Prepared for and/or by an IBM Attorney - IBM Confidential

Created By: Dave Luick **Created On:** 07/25/2001 10:55:04 AM
Last Modified By: Dan Sucher **Last Modified On:** 08/15/2001 05:07:27 PM

Required fields are marked with the asterisk (*) and must be filled in to complete the form .

***Title of disclosure (in English)**

Mechanism to avoid L0 Cache Miss and value Mispredict Pipeline Penalties

Summary

Status	
Processing Location	
Functional Area	
Attorney/Patent Professional	
IDT Team	REDACTED
Submitted Date	
Owning Division	
Incentive Program	
Lab	
Technology Code	
PVT Score	

Inventors with a Blue Pages entry

Inventors: Dave Luick/Rochester/IBM

Inventor Name	Inventor Serial	Div/Dep	Inventor	anager Name
> Luick, David A (Dave)			REDACTED	

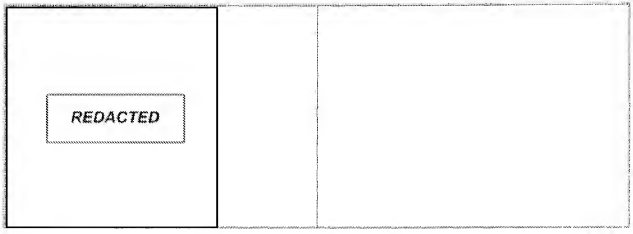
> denotes primary contact

Inventors without a Blue Pages entry

IDT Selection

Select Functional Area

IDT Team:		Attorney/Patent Professional:
	REDACTED	



Response Due to IP&L : 09/15/2001

***Main Idea**

1. Describe your invention, stating the problem solved (if appropriate), and indicating the advantages of using the invention.

A number of schemes have been proposed to initiate load instructions speculatively in advance or to build a special fast load buffer (L0) data cache to access a cache element that is likely but not certain to be the correct one. These events are much more frequent than events like TLB misses, etc. For heavily pipelined machines with a deep pipe, the recovery time to restart the instruction causing an event can well be 5-10 cycles. If half of the load instructions can make use of such speculation but the speculative fast access is wrong 20-40% as is typical for an L0 cache, the total penalty cycles can equal the speculative gain in cycles so that no or only a very small net performance gain is possible, as illustrated in Figure 1.

Pipeline Recirculation for Data Misprediction

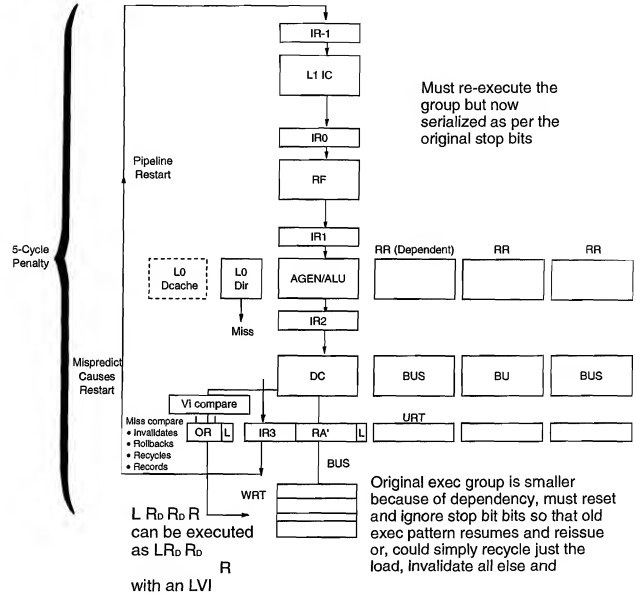


Figure 1.

However, if it is found that the speculative versions of the instructions are invalid because the speculative load instruction that initiated that instruction sequence accessed incorrect data because of a wrong "guess" access, then the speculative (and faster) load and following instruction sequence is invalidated and the nonspeculative but always correct and later in time load and its following instruction sequence is marked valid and completes execution. Thus, on a speculative "bad guess" case of a load, no time is actually lost because the nonspeculative sequence executes on time, assuming sufficient resources are always available. This is almost always the case as long as only one speculative load is allowed per cycle and two load/store and four ALU (FX) execution units are available.

2. How does the invention solve the problem or achieve an advantage,(a description of "the invention", including figures inline as appropriate)?
Figure 2 illustrates a new pipeline mechanism that can overcome the long pipeline restart problem. Basically what is done is to initiate a specially flagged (as temporarily valid) speculative or uncertain access from an L0 cache or value prediction cache which typically produces a load target result 1-3 cycles earlier than the L1 cache. Dependent instructions after the load may follow.

However, in addition, a nonspeculative load if also put done the same pipeline and also any dependent instructions on the next few cycles afterward (where the number of cycles equals the difference in latency between the L0 and L1 accesses). If it is determined that the fast speculative load instruction produced correct results, the speculative versions of the instructions are kept along with their results while the nonspeculative but now superlative versions of the instructions earlier in the pipe are invalidated and thus have no lasting effect.

Figure 3B shows an example processor core chip area floor plan that puts the LBUF near the ALUs (4) to meet the requirement that the LBUF must access and forward in one cycle just as the ALUs do. Special care is taken to especially place the RAS predecode byte (actual 6 bits). This allows the most critical RAS address bits to be very close to the AGEN logic to initiate the next cache access (LBUF/L0 or L1) quickly.

Load Buffer Physical Layout

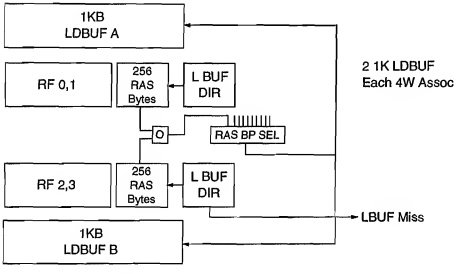


Figure 3A. Logical Load Buffer Miss

REDACTED

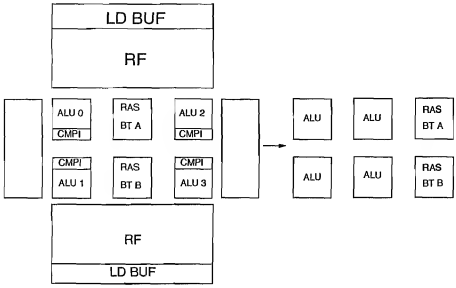


Figure 3B. Physical Layout

However, no matter how fast the LBUF (L0 DC) miss signal is made to be during the AGEN cycle of the load, it is never fast enough to allow the pipeline to execute a simple 1-cycle stall to allow the data from the L1 data cache to be used since the LBUF (L0 DC) has missed. In fact, a 2-cycle stall is difficult. Losing three cycles for every LBUF (L0) miss at a 25-30% miss rate for the LBUF (L0) eliminates most of the performance gain on hits (one cycle) so as to render it unusable. However, as the timing diagram of Figure 4 shows, there is ample time to invalidate an instruction before its target(s) is (are) written to the register file, etc. Here, the hit/miss indication from the LBUF (L0) directory can have up to two cycles to reach the register file to inhibit a write. However, it is more convenient to simply append a valid bit to each instruction and keep it as a flag bit through all stages of execution. Then, only this single valid bit need be reset (to indicate an invalid instruction) and which will prohibit any further cycles from producing results based on the corresponding instruction.

Thus, it is not only unnecessary to attempt to stall hundreds of latches throughout the pipeline but only a single latch (valid bit), but further, the remainder of the next cycle is available to actually prevent the now invalid pipe stage result from being latched (kept).

The example in Figure 2 shows the addition of an instruction reissue buffer which contains the dependent instruction immediately following the load which speculatively accesses the LBUF (L0). The instruction (R) is issued first assuming an LBUF (L0) cache hit, and then a second time N cycles later, both marked temporarily valid where N equals the number of cycles of additional latency incurred in an L1 Dcache access. As long as the LBUF (L0) hit/miss indication is known at least one cycle before the register file is written, the proper copy of the instruction (R or R') is validated and the other (R' or R) is invalidated depending on an LBUF (L0) hit or miss. Thus, both possible outcomes (LBUF hit or L1 hit) have pipelined execution in order with only the fastest valid instruction copy actually completing execution.

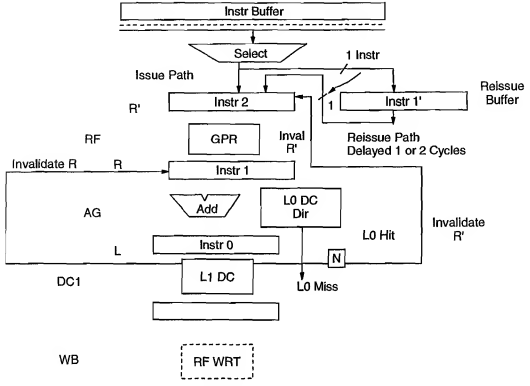


Figure 2. Speculative and Nonspeculative Instruction Reissue

Figure 4 shows a detail of both issue instances of a typical dependent instruction R. R is issued speculatively as R' assuming an L0 Dcache hit and also issued a second time as R' assuming an L0 Dcache miss/L1 Dcache hit. Both R and R' are marked temporarily valid until the L0 miss/hit indication keeps just one of them, marking the other invalid, which ultimately NOOPs the instruction by preventing its results from being clocked or used.

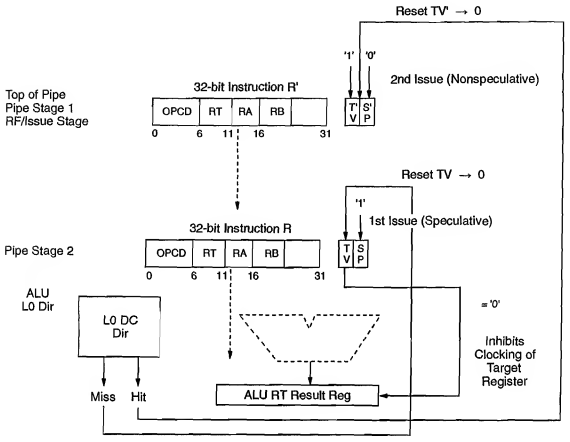


Figure 4. Temporarily Valid and Speculative Bits Appended to Each Pipeline Instruction After a Load

Single Issue/Pipeline Sequence

Cycle 1	Load (L)
Cycle 2	ADD (R) dependent on load target
Cycle 3	ADD' (R') reissue ADD, mark temporarily valid Invalidate ADD' if L0 hit Invalidate ADD if L0 miss, validate ADD'

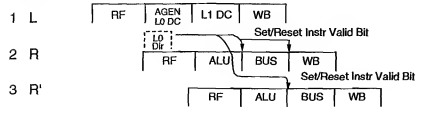


Figure 5. Example Instruction Sequence and Timings

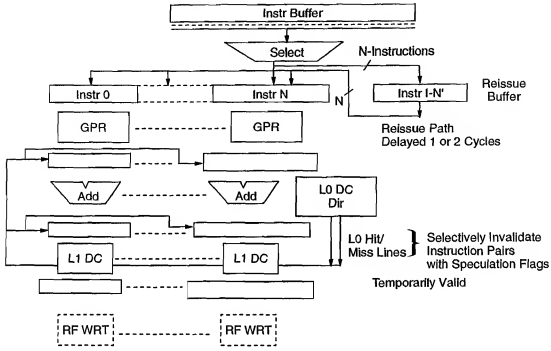


Figure 6. Generalized Multipipeline Instruction Reissue

Figure 6 shows a more generalized N-issue superscalar pipeline example where multiple instructions are issued each cycle and where multiple speculative instructions exist in the pipelines (instructions following the load) along with other nonspeculative instructions. Here, selective invalidation of instructions in pipeline stages must occur; ie, only instruction pairs marked temporarily valid are affected by LBUF (L0) miss signals where other instructions started in order without speculation are allowed to proceed normally. Here, the reissue buffer is enlarged to perhaps four instructions equaling the largest issue group for a four-pipeline superscalar. It should also be noted that the same exact scheme works to eliminate the stall conditions for value prediction miscompares with the assumption that the miscompare must be known before the end of the L1 Dcache access to invalidate a register file write in time.

3. If the same advantage or problem has been identified by others (inside/outside IBM), how have those others solved it and does your solution differ and why is it better?

4. If the invention is implemented in a product or prototype, include technical details, purpose, disclosure details to others and the date of that implementation.

***Critical Questions (Questions 1-9 must be answered in English)**

***Question 1**
On what date was the invention workable? 07/15/2001 **Please format the date as MM/DD/YYYY**
(Workable means i.e. when you know that your design will solve the problem)

***Question 2**
Is there any planned or actual publication or disclosure of your invention to anyone outside IBM?
○ Yes
● No

REDACTED

Exhibit B

3605 Highway 52 North
Rochester, MN 55901

Office of Counsel,
Intellectual Property Law
voice 507-253-2761
fax 507-253-2382

January 15, 2002

Mr. Bryan W. Bockhop
Bockhop & Reich, LLP
3235 Satellite Boulevard
Building 400, Suite 300
Duluth, GA 30096

Subject: IBM Patent Application - Docket ROC920020009US1

Dear Bryan,

Enclosed, please find a copy of the original invention disclosure and inventor information sheet for the subject IBM docket. I would like you to prepare a patent application for this invention by the workplan date of May 17, 2002, and limit preparation costs to $ REDACTED

Please feel free to call me with any comments or questions you may have. Thank you for your assistance in the preparation of this patent application.

Sincerely,

REDACTED

Enclosures

RRW/nl

PREPARED BY IP LAW REDACTED REDACTED